

This Page Is Inserted by IFW Operations  
and is not a part of the Official Record

## **BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

**IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning documents *will not* correct images,  
please do not report the images to the  
Image Problem Mailbox.**

**REMARKS**

Claims 1, 6, 7, 9, 10, 15-17 and 22-32 are now pending in the present application. Claims 1, 6, 9, 10, 15, 17, 22 and 24-32 have been amended, and Claims 2-5, 8, 11-14 and 18-21 have been cancelled, herewith. Reconsideration of the pending claims is respectfully requested.

**I. Claim Objection**

The Examiner objected to Claim 8 due to punctuation at line 3 of such claim. As Claim 8 has been cancelled herewith without prejudice or disclaimer, this rejection is now moot.

**II. 35 U.S.C. § 112, Second Paragraph**

The Examiner rejected Claims 4, 5, 8, 9, 13, 14, 20, 21, 24, 27, 29 and 30 under 35 U.S.C. § 112, second paragraph as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention. This rejection is respectfully traversed.

With respect to Claims 4, 13 and 20, the Examiner states the limitation "the installation" lacks proper antecedent basis. As Claims 4, 13 and 20 have been cancelled herewith without prejudice or disclaimer, this rejection is now moot.

With respect to Claims 5, 14 and 21, the Examiner states the limitations "the modification" and "the contents" lacks proper antecedent basis. As Claims 5, 14 and 21 have been cancelled herewith without prejudice or disclaimer, this rejection is now moot.

With respect to Claim 8, the Examiner states the limitations "the deletion" and "the plurality of environment variables" lacks proper antecedent basis. As Claim 8 has been cancelled herewith without prejudice or disclaimer, this rejection is now moot.

With respect to Claim 9, the Examiner states the limitation "the correct" lacks proper antecedent basis. Applicants have amended such claim to correct this antecedent basis issue.

With respect to Claims 24, 27 and 30, the Examiner states the limitations "the following" and "the occurring" lacks proper antecedent basis. Applicants have amended such claims to correct this antecedent basis issue.

With respect to Claim 29, the Examiner states the limitation "the following" lacks proper antecedent basis. Applicants have amended such claim to remove this objectionable terminology.

Therefore, the rejection of Claims 4, 5, 8, 9, 13, 14, 20, 21, 24, 27, 29 and 30 under 35 U.S.C. § 112, second paragraph has been overcome.

### III. 35 U.S.C. § 102, Anticipation

A. The Examiner rejected Claims 1, 10 and 17 under 35 U.S.C. § 102(e) as being anticipated by Snow (US 6,640,317). This rejection is respectfully traversed.

Applicants have amended Claims 1, 10 and 17 herewith to include features from originally filed Claim 2 (which, along with Claims 11 and 18, is thus being cancelled herewith), and to make clear that the environment variable for which the path sequence is being corrected is a currently enabled environment variable within the data processing system. It is shown that the cited reference does not teach or suggest the claimed feature of determining whether any duplicate files exist in any of the directories *identified by the path sequence* and responsive thereto, altering the path sequence. Rather, the cited Snow reference is directed to detecting changes to a computer system, comparing these changes against a set of predefined system constraints defined for each installed application, and if a discrepancy is found, taking corrective action. While similar overall objectives may exist between the cited reference and the present application, the detailed implementation as recited in Claim 1 is not taught or suggested by the cited Snow reference. In particular, Snow is concerned with maintaining proper characteristics for a given application (Snow Col. 5, lines 48-60), i.e. it is application-centric. Specifically, a determination is made as to whether any inconsistencies exist from what is required of a given, installed application program (Col. 2, lines 17-28 and lines 52-58). In contrast, the claimed invention is concerned with the overall computer system environment – i.e. it is application-neutral. Specifically, a determination is made as to whether any directories as identified by a system parameter (specifically the path sequence) have duplicate files. Therefore, it is shown that amended Claim 1 is not anticipated by the cited reference.

Therefore, the rejection of Claims 1, 10 and 17 under 35 U.S.C. § 102(e) as being anticipated by Snow has been overcome.

B. The Examiner rejected Claims 1, 10, 17, 24, 27 and 30 under 35 U.S.C. § 102(e) as being anticipated by Spyker et al. (US 6,571,389). This rejection is respectfully traversed.

Applicants have amended Claims 1, 10 and 17 herewith to include features from originally filed Claim 2 (which, along with Claims 11 and 18, is thus being cancelled herewith), and to make clear that the environment variable for which the path sequence is being corrected is a currently enabled environment variable within the data processing system. The cited reference does not teach or suggest the claimed method for correcting a path sequence of an environment variable in a data processing system, or the claimed feature of "responsive to detection of the change effecting the path sequence of the environment variable, determining whether any duplicate files exist in any of the directories identified by the path sequence". Rather, Spyker teaches that if a change is to be made to a classpath variable, the change is merely appended to the variable without further concern (Col. 14, lines 62-63).

With respect to Claim 24 (and similarly for Claims 27 and 30), Applicants show that the cited reference does not teach or suggest the combination steps of (1) automatic invocation of an environment variable manager whenever any of triggering events a), b), or c) occur, and (2) then using this same environment variable manager to determine if such triggered event causes a modification to an affected path sequence. In rejecting Claim 24, the Examiner cites Spyker's Abstract and Col. 15, lines 27-63 and FIG. 8 as teaching both the claimed (1) automatic invocation step and (2) the determining step. Applicants show that while the abstract does appear to mention dynamically changing the run-time that will be used without requiring user intervention, there is no mention or suggestion that this dynamic changing is automatically invoked whenever one of the above enumerated triggering events a), b) or c) occurs. Nor does the cited passage at Col. 15, lines 27-63 overcome this teaching deficiency. There, Spyker states:

"The run-time environment for an application can be easily changed using the present invention, according to the logic depicted in FIG. 8. At Block 800, **user input is entered from a graphical user interface (GUI), command line, etc., requesting to change information in the registry**

file. (Alternatively, means may be provided with which a systems administrator can force information updates on one or more client machines. For example, if a new run-time environment is being downloaded throughout an organization, the systems administrator may update all client registry files to use this new run-time. This approach will be useful to further reduce the amount of run-time knowledge required for the end users. Means for downloading information from a network location to client machines are known in the art, and will be used to invoke the logic of FIG. 8.) If the user request is to update the current run-time entry 645 in the registry, then Block 805 will accept the new run-time identifier from the user. Optionally, verification of this identifier may be performed. If the user request is to change persistently-stored application parameters 655, Block 810 will accept the new parameter values. Optionally, the parameter values may be verified by inspecting the applicable application to ensure that the number and type of parameter values is appropriate. If the user requests to change or add dependency information 650, then Block 815 will accept the new information. Optionally, the stored list of dependency identifiers may be presented to the user, along with means for identifying additions, deletions, and changes to this list. Once the user has entered the changed registry information, and any optional verifications have been performed, Block 820 updates the stored registry information for this application. The next time this application is launched, the revised information will be used when constructing the execution environment according to FIG. 7. Thus, it can be seen that changing an application program so that it uses a different run-time environment is greatly simplified as contrasted to the current art." (emphasis added by Applicant)

As can be seen, this passage mentions that either a user manually requests changes to a registry file, or a system administrator can force information updates on one or more client machines. There is no teaching or suggestion of automatic invocation when a directory is deleted, product is uninstalled or a path sequence is manually modified. While the above cited passage does mention manual modification of registry entries, it is shown that Claim 24 goes further, and in addition recites that a determination is made on whether the triggering event (directory deletion, product uninstall, manual environmental variable modification) causes a modification to an affected path sequence, and if so correcting the affected path sequence. This claimed feature advantageously allows for automatic correction of modifications to the affected path sequence. The cited reference merely teaches an ability to modify, but no subsequent correction of any such

modifications. Thus, it is shown that amended Claim 24 (and similarly for Claims 27 and 30) is not anticipated by the cited Spyker reference.

Therefore, the rejection of Claims 1, 10, 17, 24, 27 and 30 under 35 U.S.C. § 102(e) as being anticipated by Spyker et al. has been overcome.

C. The Examiner rejected Claims 8, 26, 29 and 32 under 35 U.S.C. § 102(e) as being anticipated by Brundridge (US 6,279,109). This rejection is respectfully traversed.

With respect to Claim 8, such claim has been cancelled herewith without prejudice or disclaimer.

With respect to Claim 26 (and similarly for Claims 29 and 32), the cited reference does not teach or suggest a determination of whether a directory specified by *a path sequence of an environment variable* is deleted. Applicants have amended Claim 26 to further clarify what is meant by this claimed path sequence. In rejecting Claim 26, the Examiner states Brundridge teaches detecting the deletion of a directory from the data processing system. Applicants urge that the amendment to Claim 26 now clearly shows that this claim is not merely directed to directory deletion detection, but rather to detecting deletion of a directory that is specified by a path sequence in an environment variable. This claimed feature advantageously provides a more focused determination to assist in managing environment variables, and in particular to managing the path sequence of an environment variable. The cited reference only contemplates changing the configuration of an operating system stored on a bootable device to point to its own root storage device, by changing a drive pointer of the operating system registry (Col. 6, lines 59-67). Changing of drive nomenclature as taught by the cited reference does not teach or suggest changing the claimed path sequence. Thus, it is shown that amended Claim 26 (and similarly for Claims 29 and 32) is not anticipated by the cited Brundridge reference.

In addition, because Brundridge's teachings are directed to creating a bootable, standalone operating system compact disk (CD), there is no necessary correlation between the specific directory structure of the underlying computer system and the directory structure of the CD operating system. Thus, there would be no motivation to modify a path sequence of the computer system when building the bootable CD (which

by its very nature is intended to be bootable, and independent from, the particular resident operating system and associated parameters/variables of the underlying computer system). Thus, it is further shown that it would not have been obvious to modify the teachings of Brundridge in accordance with the claimed invention due to such lack of motivation.

Therefore, the rejection of Claims 8, 26, 29 and 32 under 35 U.S.C. § 102(e) as being anticipated by Brundridge has been overcome.

D. The Examiner rejected Claims 9, 24, 25, 27, 28, 30 and 31 under 35 U.S.C. § 102(e) as being anticipated by Hove et al. (US 6,564,369). This rejection is respectfully traversed.

With respect to Claim 9, Applicants have amended such claim to clarify that it is directed to the *correction of modifications made* during software installation. Generally speaking, the cited Hove reference teaches a prevention technique which prevents erroneous modifications from ever being made. Because of such prevention of errors, there is no reason or other motivation to *correct modifications* that were made during software installation, as claimed. Specifically, amended Claim 9 recites a step of installing software on the data processing system. The claim goes on to recite "detecting that an environment variable has been modified during the installing step". In contrast, the cited Hove does not teach any type of detection of environment variable modification during a software installation. Rather, it *detects potential conflicts prior to being installed*, and generates a user report for manual reconciliation. Of significance is Hove's use of a configuration image file, which is defined by Hove's abstract to be "a single file containing all the information needed to troubleshoot, distribute or delete software from one or multiple computers", "the individual items in a configuration image file are entries", and "a conflict exists when one or more configuration image files include entries that are inconsistent". Thus, this is not operational software installed and executable on a computer, but rather is software capable of being installed. Hove expressly teaches a desire to check for conflicts prior to installation, as can be seen in Hove's summary (Col. 12, lines 19-21, where it states "Conflict checking functions comprise functions to check for potential conflicts in one or more configuration image files" (emphasis added by

Applicants). Hove's conflict prevention prior to installation is diametric to the claimed correction of modifications made during installation, as claimed. In addition, because of such conflict prevention, there would not have been any motivation to modify Hove's teachings in accordance with the claimed correction of modifications made during a software install. Thus, in addition to not being anticipated by the cited reference, it is further shown that it would not have been obvious to modify Hove's teachings in accordance with the claimed invention. Therefore, the rejection of Claim 9 has been overcome.

With respect to Claim 24 (and similarly for Claims 27 and 30), the cited Hove reference does not teach automatic invocation of an environment variable manager which is used to determine if the event which triggered its invocation causes a modification to *an affected path sequence*. Claim 24 has been amended to emphasize the nature of the path sequence that is modified, and in particular a determination is made on whether the occurring event, which automatically invokes the environment variable manager, cause a modification to an affected path sequence of any presently active environment variable. In contrast, the cited reference does not teach or suggest such modification of an affected path sequence, as the configuration image file that is checked for inconsistencies is not installed for execution and thus does not have presently active environment variables. While it is possible, according to Hove, to compare this configuration image file with an existing software installation (Col. 8, lines 20-22), there is no teaching or suggestion of any type of correcting of a path sequence for this existing software install. Rather, the cited reference merely teaches generation of a conflict report (Col. 8, lines 40-43). This deficiency in teaching a correction of an affected path sequence for a presently active environment variable can also be seen at Hove Col. 11, lines 35-46, where all the corrective functions listed therein are with respect to the configuration image file (and not the existing software install). There is no presently active environment variable with respect to the configuration image file, thus no affected path sequence of any presently active environment variable, and thus no modification to an affected path sequence of any presently active environment variable that is determined, as claimed. Thus, it is shown that amended Claim 24 (and similarly for Claims 27 and 30) is not anticipated by the cited reference.



With respect to Claim 25 (and similarly for Claims 28 and 31), the cited Hove reference does not teach automatic invocation of an environment variable manager whenever *a path sequence is modified*. Rather, Hove teaches manual user-initiation of conflicts checking (Col. 4, lines 42-47). Applicants have also amended Claim 25 similar to Claim 24, and further traverse the rejection of Claim 25 for similar reasons to those given above regarding Claim 24. Thus, it is shown that amended Claim 25 (and similarly for Claims 28 and 31) is not anticipated by the cited reference.

Therefore, the rejection of Claims 9, 24, 25, 27, 28, 30 and 31 under 35 U.S.C. § 102(e) as being anticipated by Hove et al. has been overcome.

E. The Examiner rejected Claims 1, 2, 5, 10, 11, 14, 17, 18, 21, 24, 27 and 30 under 35 U.S.C. § 102(b) as being anticipated by Janniro et al. (US 5,634,098). This rejection is respectfully traversed.

Applicants have amended Claims 1, 10 and 17 herewith to include features from originally filed Claim 2 (which, along with Claims 11 and 18, is thus being cancelled herewith). The cited reference does not teach or suggest the claimed method for *correcting* a path sequence of an environment variable in a data processing system, or determining whether any duplicate files exist in any of the directories identified by such path sequence. The cited Janniro reference teaches a software test system. While it is possible, as a part of a given test, to change the environment variables of the system, these environment variables are changed unconditionally as specified in the environment file (Col. 11, lines 4-19). Thus, while it may be possible to alter/change a path sequence per the teachings of Janniro, such alteration or change is not responsive to determining that duplicate files exist, as there is no teaching of any determination of such duplicate file existence. Rather, Janniro teaches any such changes are unconditional (as they are unconditional specified in the environment file). Therefore, amended Claim 1 is shown to not be anticipated by the cited reference.

Claims 2, 5, 14, 18 and 21 have been cancelled herewith without prejudice or disclaimer.

With respect to Claim 24 (and similarly for Claims 27 and 30), Applicants show that the cited Janniro reference does not teach the claimed step of "determining, by the

environment variable manager, if the occurring event causes a modification to an affected path sequence of any presently active environment variable, the path sequence specifying an order for searching directories for locating executable code within the data processing system". While it may be possible to alter/change a path sequence per the teachings of Janniro, there is no determination of *whether an occurring event causes such a change*, as Janniro teaches any such change is unconditional (as they are unconditional specified in the environment file). This claimed feature advantageously allows for correcting the affected path sequence if it is determined that the occurring event causes the modification. The cited Janniro reference does not teach any such determination, or the resulting correction to the affected path sequence. Therefore, Claim 24 (and similarly for Claims 27 and 30) is shown to not be anticipated by the cited reference.

Therefore, the rejection of Claims 1, 2, 5, 10, 11, 14, 17, 18, 21, 24, 27 and 30 under 35 U.S.C. § 102(b) as being anticipated by Janniro et al. has been overcome.

#### IV. 35 U.S.C. § 103, Obviousness

The Examiner rejected Claims 3, 4, 6, 7, 12, 13, 15, 16, 19, 20, 22 and 23 under 35 U.S.C. § 103 as being unpatentable over Janniro et al. (US 5,634,098) in view of Hove et al. (US 6,564,369). This rejection is respectfully traversed.

With respect to Claims 3, 4, 12, 13, 19 and 20, such claims have been cancelled herewith without prejudice or disclaimer.

With respect to Claims 6 and 7 (and similarly for Claims 15, 16, 22 and 23), Applicants traverse for reasons given above regarding Claim 1.

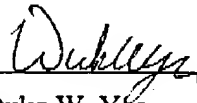
Therefore, the rejection of Claims 3, 4, 6, 7, 12, 13, 15, 16, 19, 20, 22 and 23 under 35 U.S.C. § 103 has been overcome.

**V. Conclusion**

It is respectfully urged that the subject application is patentable over the cited references and is now in condition for allowance. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: 7/1/04

Respectfully submitted,



Duke W. Yee  
Reg. No. 34,285  
Wayne P. Bailey  
Reg. No. 34,289  
Yee & Associates, P.C.  
P.O. Box 802333  
Dallas, TX 75380  
(972) 367-2001  
Attorneys for Applicants